# SINGLE VIDEO PERFORMANCE ANALYSIS FOR VIDEO-ON-DEMAND SYSTEMS

BY

JAMES YIFEI YANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Adviser:

Professor Bruce Hajek

# ABSTRACT

We study the content placement problem for cache delivery video-on-demand systems under static random network topologies with fixed heavy-tailed video demand. The performance measure is the amount of server load; we wish to minimize the total download rate for all users from the server and maximize the rate from caches. Our approach reduces the analysis for multiple videos to consideration of decoupled systems with only a single video. For each placement policy, insights gained from the single video analysis carry back to the original multiple video content placement problem. Finally, we propose a hybrid placement technique that achieves near optimal performance with less complexity.

# TABLE OF CONTENTS

# CHAPTER 1

# INTRODUCTION

A video-on-demand (VoD) system is an online video delivery system in which peers can request which videos to watch. In order for peers to watch videos without interruptions or large delays, the system must meet stringent delivery requirements - peers need to begin to download quickly and stream at the playback rate.

Traditionally, all video requests in a VoD system were handled by a central server. However, as the number of videos and peers grow, it becomes increasingly difficult for one central server to provide all the storage and bandwidth. Moreover, due to the increasing spread of peer locations, the size of the network grows and cost of delivery increases. Therefore, it is reasonable to design a cache delivery system in which each cache acts as a small server to help the central server and is placed at the center of a cluster of peers. Each peer is connected to a subset of the caches in close proximity and when a peer requests a video to watch, it sends out requests to all of its connected caches. However, if the connected caches do not have the requested video in storage or the rate of upload from caches is insufficient, then the peers seek the server for help for the missing parts. Therefore, a reasonable performance measure is the amount of server load.

In our analysis, we primarily study the content placement problem: Given a set of caches, what set of videos should be stored at each cache under storage constraints, current demand, and network topology? There are two types of storage methods for this problem: **1.** Whole storage - videos are stored as whole copies, and **2.** Fractional storage - with the help of source codes such as maximum-distance-separable (MDS) codes, videos are coded and stored as fractions of a copy. When a peer requests a video, the whole storage architecture requires the peer to download only from one of its connected caches containing the video. In the fractional storage architecture, a peer can simultaneously download from all of its connected caches containing the coded fractions of the video, and the download rates are summed by the additivity property of the MDS code.

Under the aforementioned storage methods, placement policies can be categorized into adaptive and fixed (non-adaptive) placement. Under adaptive placement policies, copies of a particular video are stored in the set of caches with the most received requests. Under fixed placement policies, caches are oblivious to the current demand so the random placement of copies of a particular video has the same statistics as the deterministic placement of the copies.

Although adaptive fractional storage placement provides the upper bound on performance because it minimizes the server load and is distributed, it is inefficient in the sense that it requires overhead and computational power to encode and decode the videos, resulting in delays. In contrast, optimal adaptive whole storage placement is optimal under integer programming, which is combinatorial in nature and becomes computationally intractable when the system grows large. While fixed fractional storage placement is simple to implement and has linear performance, all caches store some fractions of the same video regardless of the actual connectivity. Finally, fixed whole storage placement, based on the uniform random assignment, provides the lower bound on performance. The objective of this work is to compare these four content placement options. We also propose a hybrid placement technique that achieves near optimal performance with less complexity.

# CHAPTER 2

# RELATIONSHIP WITH RELATED WORK

VoD cache delivery systems have received wide attention for their benefits - the reduction of content delivery cost and the improvement of the end-user performance. Popular video sharing websites such as Youtube have been aggressively deploying cache servers of widely varying sizes at many locations around the world [1]. In addition, cache servers in cache delivery systems appear as television set top boxes or personal computers in peer-to-peer networks like Akamai [2] and PPLive [3]. There are a number of works on content placement [4, 5, 6, 7, 8, 9, 10, 11]. Almeida et al. [4] studied content placement and routing optimization in cache delivery systems subject to path delivery cost under a fixed topology. Boufkhad et al. [5] derived bounds on the number of videos that can be served in the system subject to storage constraint, upload constraint, and cache connection degree. B. Tan et al. [6] established an asymptotically optimal content placement strategy subject to a storage constraint and loss network model. Zhou et al. [7] focused on balancing the workload of caches. Laoutaris et al. [8] studied cache storage resource allocation. Since our primary focus is on the content placement problem, the problem is only subject to a given storage constraint and does not involve any path delivery cost or cache upload constraint. In our formulation, we have a fixed topology generated by randomly established cache-peer connections subject to peer connection degree. As long as any of the caches a peer is connected to stores the requested video, the peer can be served by the cache delivery system without the help of the central server.

Wu and Li [9] studied the optimal cache replacement algorithm and suggested that the simplest heuristics perform as well as the optimal algorithms, with very insignificant differences. While our formulation assumes random but static video requests, we also look for a simple suboptimal alternative to the optimal content placement algorithm. In this thought, we decompose the analysis of the content placement problem from the scale of the entire system with multiple videos into decoupled systems, each with only a single video of given popularity. Applegate et al. [10] formulated the adaptive whole storage placement problem as a mixed integer program (MIP) subject to a storage constraint and link bandwidth constraint. The adaptive whole storage placement problem is solved approximately by MIP. In our work, we derive an upper and a lower bound on the performance of the single video adaptive whole storage placement policy using analytical and heuristic approaches. Zhang [11] used MDS codes to relax the integer constraint and converted the integer program into a convex, adaptive fractional storage placement problem that can be solved exactly. This result provides an upper bound on the performance of any content placement policy, including all single video placement policies. Having this upper bound, we study the remaining single video placement policies, fixed whole storages, and fixed fractional storage placement policies. With the insights gained from single video placement policy analysis, we return to the original content placement problem by constructing a general method for the four placement policies for multiple videos, which places copies of videos optimally for each policy. Then, combined with observations from single video comparisons, we introduce a hybrid storage multiple video placement policy which is a suboptimal alternative to the adaptive fractional storage placement policy from Zhang [11].

To the best of our knowledge, we have been the first to decompose the analysis of the content placement problems for the whole system with multiple videos into decoupled systems with a single video and carry the result back to the original content placement problem.

# CHAPTER 3

# MODEL AND ASSUMPTIONS

To focus strictly on the analysis of content placement policies in the VoD cache system, we construct a simple closed homogeneous system model. We consider the server to be external to the cache system and to provide help only when caches within the system cannot satisfy all the peer demand, shown in Figure 3.1. In the model, the numbers of peers, videos, and caches remain fixed. Each peer is connected to an equal fixed number of caches. We assume peer location is uniformly distributed, so peer connections (the set of caches connected to a peer) are also uniformly distributed.

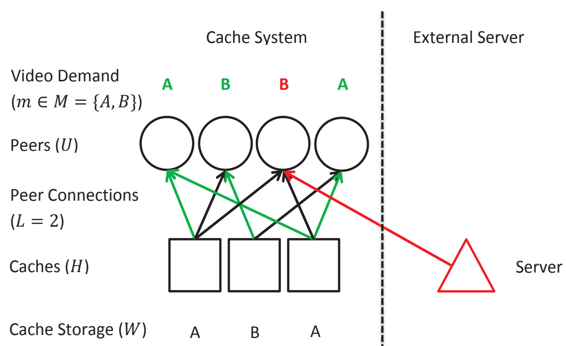


Figure 3.1: Cache system of three caches, four peers and two random connections per peer

Under this setting, we have a model of a random but fixed network topology (a graph of cache-peer connections). Each video in the system occupies the same storage space and has the same playback rate. Peers in the system watch videos continuously and independently select videos according to a heavy-tailed popularity distribution, typically a $Zipf(\text{number of videos}, 0.8)$ distribution. Based on the law of large numbers (LLN), the number of peers requesting video i at any instance is close to the expected number. Since the expected number of peers connected to each cache is the same, we assume each cache has sufficient bandwidth to provide the requested streaming rates to all connected peers. We also ignore any link bandwidth constraints, so there are no bandwidth constraints imposed by an underlay network. At any time, peers can only download as much as the required playback rates, so they do not store any future video chunks.

## 3.1 Problem Formulation

The notation of the model is summarized as follows:

- $H$: set of caches
- $U$: set of peers
- $M$: set of videos, ordered according to popularity
- $G$: set of all possible network topology graphs (i.e. peer-cache connection graph)
- $N_u^g$: set of caches connected to peer $u$ under graph $g \in G$
- $H_m$: set of caches storing video $m$
- $U_m$: set of users requesting video $m$
- $C_m$: number of copies of video $m$ stored in the cache system (possibly sum of fractional parts)
- $S_h$: storage capacity of cache $h$ in units of videos
- $Zipf(K,\ 0.8)$: probability distribution over $\{1, ..., K\}$ given by $p(m) = \frac{1/m^{0.8}}{\sum_{n=1}^{K}(1/n^{0.8})}$
- $p(m)$: video popularity distribution (i.e. probability the video requested by a peer is video $m$)
- $\alpha(m)$: probability video $m$ is present in a typical cache given by $\alpha(m) = \frac{C_m}{|H|}$
- $x_{hu}$: download rate of peer $u$ from cache $h$ (possibly a fractional rate)

- $W_{hm}$: fraction of video $m$ on cache $h$

Given the set of caches $H$ and the set of peers $U$, under a fixed network topology $g \in G$, each peer $u$ is connected to $|N_u^g| = L$ random caches selected uniformly. Given the set of videos $M$, the probability a peer requests video $m$, $p(m)$, follows Zipf($|M|$,0.8) distribution and the mean number of peers requesting video $m$ satisfies $E[|U_m|] = |U| \cdot p(m)$. Each cache $h$ has an equal storage capacity of $S_h$ units of video and stores $W_{hm}$ units of video $m$, where $W_{hm} \in \{0,1\}$ for whole storage placement policies and $W_{hm} \in [0,1]$ for fractional storage placement policies. Each peer $u$ downloads at rate $x_{hu}$ from cache h for each of its connected caches $h \in N_u^g$, where $x_{hu} \in \{0,1\}$ for whole storage placement policies and $x_{hu} \in [0,1]$ for fractional storage placement policies.

The cost function we want to minimize is the server load, or equivalently, the utility function we want to maximize is the total download rate for all users from the caches. The following optimization problem for a given graph $g = (N_u^g : u \in U)$, was formulated in [11]:

$$
\begin{aligned}
\max \quad & \sum_{u \in U} \min \left\{ \sum_{h \in N_u^g} x_{hu}, 1 \right\} && (3.1)\\
w.r.t. \quad & x_{hu} \text{ and } W_{hm} \ \forall u \in U, h \in H, m \in M \\
s.t. \quad & x_{hu} \leq W_{hm} \ \forall h, u \ and \ m : u \in U_m, \\
& \sum_{m \in M} W_{hm} \leq S_h \ \forall h, \\
& W_{hm} \in \{0,1\} \ \forall h, m \text{ (whole storage placement)}, or \\
& W_{hm} \in [0,1] \ \forall h, m \text{ (fractional storage placement)}.
\end{aligned}
$$

## 3.2 Fractional Storage and Adaptive Placement Methods

Fractional storage is a way of storing video copies that allows each copy to be split into pieces of smaller sizes and these pieces are placed in different caches. When a peer downloads from multiple connected caches that store some pieces of the video they are requesting, if every downloaded piece is useful we denote this as the additivity property. For any fractional storage method with the additivity property, the download rate of a peer is the sum of the download rates from the peer's connected caches.

One fractional storage method that has the additivity property is source coding using MDS code. Each video chunk is coded into pieces of smaller size and these pieces are placed into caches. Every piece a peer downloads is useful so any subset of the coded pieces that approximately sums to the chunk size can be used to reconstruct the original video chunk. Another fractional storage method that has the additivity property is time-sharing. Each video chunk is split into substreams through time division and these substreams are placed into caches. Only the non-redundant substreams a peer downloads are useful. The pieces of video chunks can also be viewed as video chunks of smaller size, so the placement policy is whole storage again with the whole units being the substreams. Therefore, time-sharing fractional storage methods have more constraints than source coding fractional storage methods. We will use source coding for fractional storage in this thesis.

Adaptive placement is a way of placing video copies that maximizes the total download rate for all users from the caches in response to the current demand, i.e. copies of a particular video are stored in the set of caches with the most received requests. Fixed (non-adaptive) placement is the opposite of adaptive placement in which caches are oblivious to the current demand, so the statistics of the random placement of copies of a particular video is the same as the deterministic placement of the copies.

# CHAPTER 4

# SINGLE VIDEO PLACEMENT POLICY ANALYSIS

To gain insight into the multiple video content placement policies for the whole system, we decompose the analysis of the content placement problem into decoupled systems with a single video. We will first discuss single video placement, in which each content placement policy is analyzed for a single video $m$ with arbitrary popularity.

## 4.1 Fixed Whole Storage Placement Policy

For fixed whole storage placement policies, caches do not change their video catalogs according to the actual demand. Given an integer number of video copies, $C$, to be placed in the cache system, the set of caches of cardinality $C$ is randomly selected to store one whole video copy in each cache; then, $|H_m| = C$. A peer$u$ can download from at most one of its $L = |N_u^g|$ connected caches that store the requested video. Let $p_{miss}(m)$ denote the probability a peer is not connected to a cache storing video $m$, given by $p_{miss}(m) = \frac{\binom{|H|-C}{L}}{\binom{|H|}{L}}$. Given the set of peers, $U$, requesting video $m$, the expected number of peers that are served by the cache system without help from the server is given by:

$$
\begin{aligned}
& E[\text{total download rate provided by caches}] \\
= \; & E[\#\text{ of peers requesting video } m \text{ served by caches}] \\
= \; & (\#\text{ of peers requesting video } m) \cdot \\
& \qquad P\{\text{a peer is connected to at least 1 cache storing video } m\} \\
= \; & |U| \cdot (1 - p_{miss}(m)) \qquad (4.1)
\end{aligned}
$$

The fixed whole storage placement policy provides a good benchmark for all single video placement policies, because adaptive placement policies and fractional storage placement policies, discussed below, are generalizations of fixed whole storage placement policies. Adaptive placement policies can be made to perform better than fixed placement policies because caches see the actual demand and their video catalogs change accordingly. Also, fixed fractional storage placement policies can be made to perform better than fixed whole storage placement policies, which will be shown in the next section, 4.2.

In view of Equation (4.1), Proposition 4.1.1 provides a simple lower bound on the expected total download rate for video $m$ provided by the caches. Let $\alpha(m)$ be the probability video $m$ is present in a typical cache for fixed whole storage placement. In general, the mean number of video copies $C$ to be placed in the cache system, $\alpha(m) \cdot |H|$, can be any real number, so we write $\alpha(m) \cdot |H| = \lfloor C \rfloor + \theta$, where $0 \leq \theta < 1$. Let $X$ be the minimum variance integer valued random variable with mean $\alpha(m) \cdot |H|$. Specifically, $P\{X = \lfloor C \rfloor\} = 1 - \theta$ and $P\{X = \lfloor C \rfloor + 1\} = \theta$. Given $X$, the video is assumed to be placed into a set of caches of cardinality $X$, with all $\binom{|H|}{X}$ possibilities having equal probability. We will provide a bound on $p_{miss}(m)$ with the following proposition:

**Proposition 4.1.1.** $p_{miss}(m) \leq (1 - \alpha(m))^L$.

*Proof.* On one hand, factoring out common terms yields:

$$\begin{aligned} p_{miss}(m) &= (1-\theta)\left(\frac{n-k}{n}\right)\left(\frac{n-k-1}{n-1}\right)\cdots\left(\frac{n-k-L+1}{n-L+1}\right) \\ &\quad + \theta\left(\frac{n-k-1}{n}\right)\left(\frac{n-k-2}{n-1}\right)\cdots\left(\frac{n-k-L}{n-L+1}\right) \\ &= \frac{(n-k-\theta L)(n-k-1)\cdots(n-k-L+1)}{n(n-1)\cdots(n-L+1)} \end{aligned} \tag{4.2}$$

On the other hand, using the convexity of $(a - b\theta)^L$ as a function of $\theta$, and the fact $f(\theta) \geq f(0) + \theta f'(0)$ for a convex function $f$,

$$\begin{aligned} (1-\alpha(m))^L &= \left(\frac{n-k-\theta}{n}\right)^L \\ &\geq \left(\frac{n-k}{n}\right)^L - \frac{\theta L}{n}\left(\frac{n-k}{n}\right)^{L-1} \\ &= \left(\frac{n-k-\theta L}{n}\right)\left(\frac{n-k}{n}\right)^{L-1} \end{aligned} \tag{4.3}$$

Comparing (4.2) to (4.3) completes the proof of the lemma. □

The single video performance of the fixed whole storage placement policy given by Equation (4.1) and its lower bound are plotted in Figure 4.1 for a video $m$ requested by 20 peers (i.e. $|U| = 20$). The expected total download rate provided by the caches is plotted versus the number of copies of the video stored in the cache system. It can be seen that, due to the randomness in the fixed whole storage placement policy, nearly 3/4 of the caches need to store a copy of the video in order for every peer to be served by the cache system without help from the server. If $L << |H|$, then whether one of the caches connected by the peer has the video is approximately independent of whether the other connected caches have the video. Therefore, $p_{miss}(m) \approx (1 - \alpha(m))^L$, so the two curves on the plot are nearly identical.

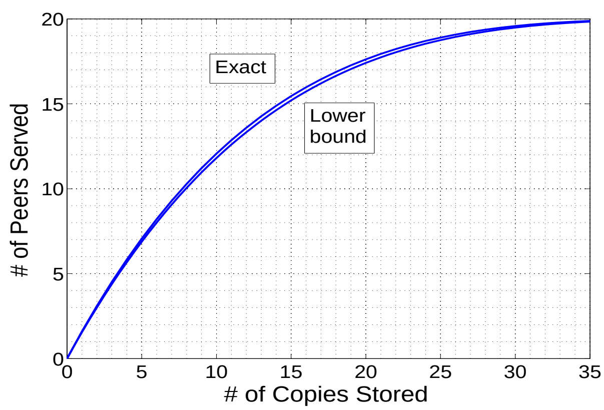


Figure 4.1: Single video performance of a cache system of 50 caches, 20 peers and four random connections per peer

## 4.2 Fixed Fractional Storage Placement Policy

For fixed fractional storage placement policies, caches' video catalogs also remain fixed regardless of the demand. For the fixed uniform fractional storage placement policy, with a given number (possibly non-integer) of video copies, $C$, in the cache system, each cache stores the uniform fraction, $W_h = \frac{C}{|H|}$, of video $m$. A peer can download from all of its $L = |N_u^g|$ connected caches and all peers download at the same rate. However, peers may download at a fraction of the required playback rate, so possibly none of them are fully served by the caches. Summing the download rates from all peers requesting video $m$, we find the total download rate provided by the cache system for the set of peers, $U$, without help from the server:

$$\begin{aligned}
& \text{(total download rate provided by caches)} \\
= & \sum_{u \in U} \min \left\{ \sum_{h \in N_u^g} \frac{C}{|H|}, 1 \right\} \\
= & |U| \cdot \min \left\{ L \frac{C}{|H|}, 1 \right\} \qquad (4.4)
\end{aligned}$$

Given the number of copies of video $m$, $C$, we can find the expected fraction of total download rate provided by the cache system without help from the server. Since caches are uniformly randomly selected by peers, the expected download rate served by each connection is the same for any fixed fractional storage placement. On one hand, the download rate served by each connection under the fixed uniform fractional storage placement is deterministic. As a result, for each additional video copy placed in the cache system, a peer's download rate increases linearly and deterministically. On the other hand, the download rate served by each connection under other fixed fractional storage placements is random. Therefore, as shown in the following proposition, fixed uniform fractional storage placement outperforms any other fixed fractional storage placements, in particular the fixed whole storage placement.

**Proposition 4.2.1.** *Among all fixed fractional storage placements, the uniform fractional storage placement maximizes the expected total download rate.*

*Proof.* It suffices to show the expected download rate of a peer requesting video $m$ served by $L$ random connections is maximized by the uniform fractional storage placement.

The download rate of peer $u$ from cache $h$ is given by $x_{hu} = W_{hm} \cdot \mathbb{1}_{\{h \in N_u\}}$.

Therefore,

$$\begin{aligned}
& E[\text{download rate of peer } u \text{ from caches}] \\
= \ & E\left[\min\left\{\sum_{h\in N_u^g} x_{hu}, 1\right\}\right] \\
\leq \ & \min\left\{\sum_{h\in N_u^g} E[x_{hu}], 1\right\} \text{ by Jensen's inequality} && (4.5) \\
= \ & \min\left\{\sum_{h\in N_u^g} W_h \cdot P\left\{h \in N_u^g\right\}, 1\right\} \\
= \ & \min\left\{\sum_{h\in N_u^g} W_h \cdot \frac{L}{|H|}, 1\right\} \\
= \ & \min\left\{L\frac{C}{|H|}, 1\right\} && (4.6)
\end{aligned}$$

Equality in Equation (4.5) is achieved if $\sum_{h\in N_u^g} x_{hu} \equiv \sum_{h\in N_u^g} E[x_{hu}]$, which is true for uniform fractional storage placement. □

Note that the whole storage placement is a special case of fractional storage placement where fractions are constrained to be zero or one. The performance of the fixed uniform fractional storage placement policy is plotted in Figure 4.2 for a video $m$ requested by 20 peers (i.e. $|U| = 20$). The total download rate provided by the caches is plotted versus the number of copies of the video stored in the cache system. It can be seen that, due to the linear and deterministic properties of the fixed fractional storage placement policy, the marginal performance gain of an additional video copy is always the same until the number of copies reaches the minimum number needed to serve the entire video to all peers. Every cache only needs to store a fraction, $\frac{1}{L}$, of the video in order for every peer to be served by the cache system without help from the server.

## 4.3 Adaptive Whole Storage Placement Policy

For adaptive whole storage placement policies, caches select their video catalogs in response to the actual demand. For optimal adaptive whole storage

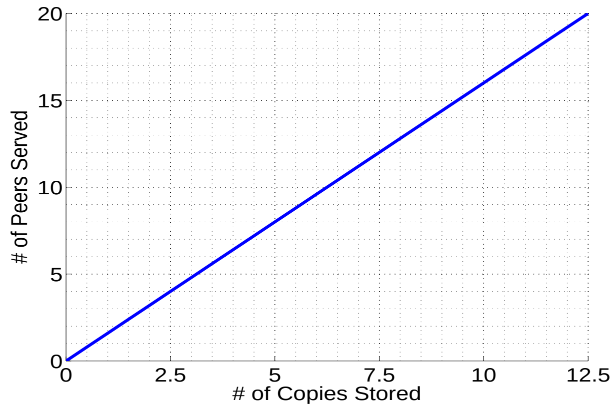


Figure 4.2: Single video performance of a cache system of 50 caches, 20 peers and four random connections per peer

placement, given an integer number of video copies, $C$, to be placed in the cache system, the set of caches of cardinality $C$ that is connected to the largest group of peers requesting video $m$ stores the video; we denote this set of caches by $H_m$. The total download rate provided by the cache system for the set of peers, $U$, without help from the server is the number of peers requesting video $m$ that are connected to $H_m$. Although finding $H_m$ is an NP-hard problem (because the set covering problem is NP-complete), we can approximate the expected number of peers requesting video $m$ that are connected to $H_m$ by giving an upper and a lower bound.

Let $p_{miss}(m)$ denote the probability a peer is not connected to a cache storing video $m$, given by $p_{miss}(m) = \frac{\binom{|H|-C}{L}}{\binom{|H|}{L}}$. We have the following upper bound:

**Proposition 4.3.1.** *E[total download rate provided by $H_m$]* $\leq \sum_{\tau=1}^{|U|} \min\left(\binom{|H|}{C} \cdot Bin^c\left(|U|, 1-p_{miss}(m), \tau-1\right), 1\right)$, *where $Bin^c(N,p,K)$ is the complementary CDF of binomial distribution at $K$ with corresponding number of trials $N$, and probability of success for each trial $p$. i.e. $Bin^c(N,p,K) = 1 - Bin(N,p,K)$. The index $\tau$ ranges over the positive integers less than or equal to $|U|$, the number of peers requesting video $m$.*

*Proof.* Consider any fixed set $A$ of cardinality $C$,

$$\begin{aligned} & P\{\text{a given peer is connected to at least one cache in } A\} \\ = \ & 1 - p_{miss}(m) \qquad (4.7) \end{aligned}$$

So, the number of peers connected to at least one cache in A has the binomial distribution with parameters $N = |U|$ and $p$ from Equation (4.7). Thus, for any integer $\tau \geq 1$,

$$P\{A \text{ covers at least } \tau \text{ peers}\} = Bin^c(|U|, 1 - p_{miss}(m), \tau - 1) \tag{4.8}$$

Let $Y$ be the number of sets of caches of cardinality $C$ that have at least $\tau$ peers connected to them.
Since there are $\binom{|H|}{C}$ sets of caches of cardinality $C$ and any such set of caches has probability $p$ to cover at least $\tau$ peers as $A$,

$$\begin{aligned} E[Y] &= \binom{|H|}{C} \cdot P\{A \text{ covers at least } \tau \text{ peers}\} \\ &= \binom{|H|}{C} \cdot Bin^c(|U|, 1 - p_{miss}(m), \tau - 1) \end{aligned} \tag{4.9}$$

By the first moment bound, for the non-negative integer random variable $Y$,

$$P\{\# \text{ of peers served by } H_m \geq \tau\} = P\{Y \geq 1\} \leq \min\{E[Y], 1\} \tag{4.10}$$

Finally,

$$\begin{aligned} & E[\text{total download rate provided by the } H_m] \\ = & \sum_{\tau=1}^{\tau=|U|} P\{\# \text{ of peers served by } H_m \text{ caches} \geq \tau\} \end{aligned} \tag{4.11}$$

Therefore, Equation (4.9)-(4.11) yields the proposition □

Next, we introduce a heuristic method for obtaining a suboptimal set of caches with cardinality $C$. The algorithm first finds a cache that is connected to the largest number of peers requesting video $m$ in each iteration and stores a copy in that cache. Then, it peels away (removes) all peers connected to that cache requesting video $m$. The algorithm is shown in Algorithm 1.

Since the greedy peeling algorithm may not choose the set of caches of cardinality $C$ that are connected to the maximum of the number of peers requesting video $m$, the number of peers connected to these caches is a lower bound on the number of peers connected to $H_m$ for any graph. Therefore, the statistical average of total download rate provided by sets of caches of cardinality $C$ chosen by the greedy peeling algorithm over random graphs is

**Algorithm 1** Greedy peeling algorithm for adaptive whole storage placement of C copies of a single video m

1: **while** not all $C$ copies of video $m$ are placed in the caches **do**
2: - find $h_{max}$ with the most connected peers requesting video $m$
3: - store a copy of video $m$ in cache $h_{max}$
4: - remove all peers connected to $h_{max}$ requesting video $m$ from the cache system
5: **end while**

a lower bound on the expected total download rate provided by the cache system without help from server.

The performance of the adaptive whole storage placement policy is plotted in Figure 4.3 for a video $m$ requested by 20 peers (i.e. $|U_m| = 20$), where the upper bound is obtained from Proposition 4.3.1 and the lower bound is obtained from Algorithm 1. The bounds on the total download rate provided by the caches are plotted versus the number of copies of the video stored in the cache system. Note that the upper bound is obtained analytically and the lower bound is obtained as an average over random graphs computed by simulation. It can be seen that, due to the adaptiveness in the adaptive whole storage placement policy, only a small portion of the caches need to store a copy of the video in order for every peer to be served by the cache system without help from the server. The decrease in the marginal performance gain follows from the random connections which result in a non-symmetric graph; some caches have more peers connected than others.

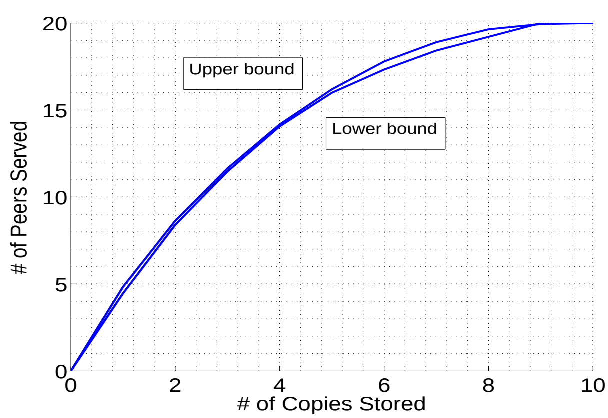


Figure 4.3: Single video performance of a cache system of 50 caches, 20 peers and four random connections per peer

## 4.4 Adaptive Fractional Storage Placement Policy

For adaptive fractional storage placement policies, caches' video catalogs are also selected in response to the actual demand. Given the number (possibly non-integer) of video copies $C_m$ to be placed in the cache system, each cache stores some fraction of video $m$, $W_{hm}$. The values $(W_{hm} : h \in H, m \in M)$ are chosen to maximize the total download rate provided by the cache system without help from the server. Since adaptive fractional storage placement relaxes the integer constraint in Equation (3.1), we can solve the convex optimization problem exactly. Primal-dual algorithm can be applied for solving the convex optimization problem. We first set the download rate of each cache-peer connection and the storage of video $m$ in each cache as primal variables. We then set the price of download rate of each cache-peer connection subject to the availability of video $m$ in the cache and the price of total cache storage of video $m$ subject to a global cache storage constraint as dual variables. Thus the cache system's fractional storage placement, routing, and upload rates converges optimally by the primal-dual algorithm [11]:

**Algorithm 2** Primal-dual algorithm for adaptive fractional storage placement of C copies of a single video m

Primal 1: update the download rates

$$\dot{x}_{hu} = \left[\delta_{hu} \cdot \left(\mathbb{1}_{\left\{\sum_{h' \in N_u^g} x_{h'u} < 1\right\}} - \lambda_{hu}\right)\right]^+_{x_{hu}} \tag{4.12}$$

where $\delta_{hu} > 0$ is an adaptation parameter

Dual 1: update the price of download rates

$$\dot{\lambda}_{hu} = [\kappa_{hu} \cdot (x_{hu} - W_h)]^+_{\lambda_{hu}} \tag{4.13}$$

where $\kappa_{hu} > 0$

Primal 2: update the cache storages

$$\dot{W}_h = \left[\iota_h \cdot \left(\left(\sum_{u: h \in N_u^g} \lambda_{hu}\right) - \omega\right)\right]^+_{W_h} \tag{4.14}$$

where $\iota_h > 0$

Dual 2: update the price of cache storages

$$\dot{\omega} = \left[\nu \cdot \left(\sum_{h \in H} W_h - C\right)\right]^+_{\omega} \tag{4.15}$$

where $\nu_h > 0$

The performance of the adaptive fractional storage placement policy is plotted in Figure 4.4 for a video $m$ requested by 20 peers (i.e. $|U_m| = 20$). The total download rate provided by the caches is plotted versus the number of copies of the video stored in the cache system. It can be seen that, due to the adaptiveness in the adaptive fractional storage placement policy, only a small portion of the caches need to store a copy of the video in order for every peer to be served by the cache system without help from the server. The concavity in the non-decreasing marginal performance gain follows from the random connections which result in a non-symmetric graph. The slower decrease in the marginal performance gain of the adaptive fractional storage placement policy compared to the adaptive whole storage placement policy resulted from the relaxation of the integer storage placement constraint.

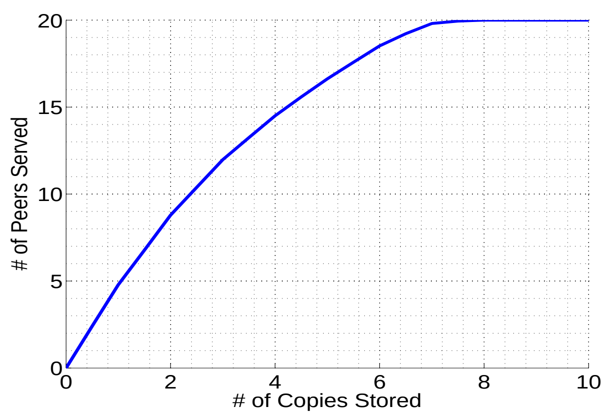


Figure 4.4: Single video performance of a cache system of 50 caches, 20 peers and four random connections per peer

# CHAPTER 5

# SINGLE VIDEO PLACEMENT POLICY COMPARISONS

This chapter explores the trade-offs between performance and practicality of the placement policies for a single video discussed in Chapter 4. First, we will illustrate the analysis results of single video placement policies for videos at three different popularity levels.

To look for potential patterns in single video placement performances, we choose 20, 50, 100, and 2000 peers representing three different popularity levels, where each peer is connected to four random caches out of the 50 caches selected uniformly. Videos with 20 or fewer peer requests represent the set of videos with below average popularity. A video with 100 peer requests represents a video of above average popularity. A video with 2000 peer requests represents the most demanded video. Analysis results obtained from Chapter 4 are shown in Figures 5.1 - 5.3. The fraction of total download rate provided by the caches is plotted versus the number of copies of the video stored in the cache system. FW, FF, AW, and AF represent the fixed whole, fixed fractional, adaptive whole, and adaptive fractional storage placement policy, respectively.

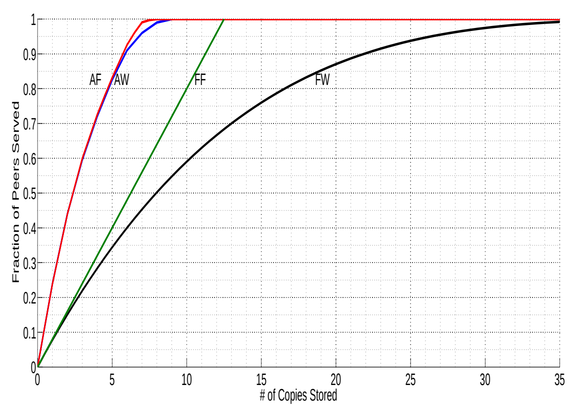


Figure 5.1: Single video performance of a cache system of 50 caches, 20 peers and four random connections per peer

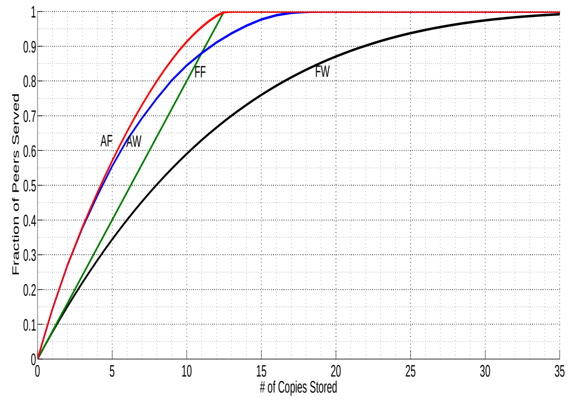


Figure 5.2: Single video performance of a cache system of 50 caches, 100 peers and four random connections per peer

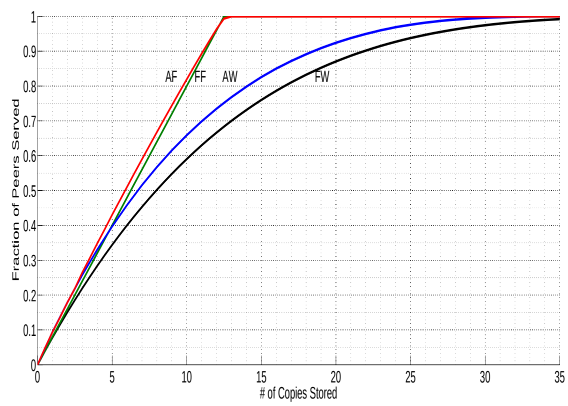


Figure 5.3: Single video performance of a cache system of 50 caches, 2000 peers and four random connections per peer

From these three plots, the first observation is that the single video placement performance curves of fixed whole and fixed fractional storage placement policies remain the same across videos of all popularity. This is true because the non-adaptive property of these two policies yields the same expected performance over random graphs.

The second observation is that the fixed whole storage placement policy serves as the performance lower bound and the adaptive fractional storage placement policy serves as the performance upper bound on all four single video placement policies. This fact is true for videos of all popularity because of the integer constraint of whole storage placement policies and the non-adaptive property of fixed placement policies. A consequence of this fact is that the performance curve of adaptive whole storage placement policy is always above that of the fixed whole storage placement policy, and the performance curve of adaptive fractional storage placement policy is always above that of the fixed fractional storage placement policy. Basically, the adaptive whole storage and fixed fractional placement policies are the only pair of curves that can cross.

The third observation is that the performance curves of adaptive whole and adaptive fractional storage placement policies drop down and converge to those of their corresponding fixed placement policies as videos become more popular. This is due to the fact that as more peers request a video, the LLN implies each cache is approximately connected to the same number of peers. As a result, the adaptive property of placement policies becomes less beneficial, which means the placement of copies of a video into any random set of caches yields approximately the same performance.

The fourth observation is that the performance curve of adaptive whole storage placement policy rises up and converges to the adaptive fractional storage placement policy as videos become less popular. This is due to the fact that as fewer peers request a video, there will be more imbalance in the cache-peer connections due to randomness. As a result, the integer constraint relaxation of the adaptive fractional storage placement policy becomes less imposing, which means the adaptive placement of whole copies of a video yields approximately the same performance.

# CHAPTER 6

# MULTIPLE VIDEO PLACEMENT POLICY AND SIMULATIONS

In this chapter, we find a general method to apply the algorithms and analysis developed for single video placement to placement of multiple videos. The general method places copies of videos optimally for each policy by maximizing the total download rate provided by the cache system without help from the server. Then, motivated by observations from single video comparisons, we introduce a hybrid storage multiple video placement policy.

The system model remains the same as the one described in Chapter 3. Given the storage capacity of each cache, the goal is to maximize the total download rate for all users from the caches. Since we are considering the placement of copies of different videos in the multiple video scenario, to simplify the problem, we will sometimes consider a single combined total cache storage constraint first. Then, we will add the uniform per-cache storage constraint and comment on the difference.

## 6.1 General Algorithm for Multiple Video Placement Policy

For each placement policy, using the algorithms and analysis developed for single video placement, we iteratively store copies of videos with the highest marginal performance gain subject to the storage constraint, shown in Algorithm 3. This method maximizes the total expected download rate provided by the cache system without help from the server. This is because placing copies of one video will not affect the marginal performance gains of other videos and the marginal performance gain of a video depends only on the numbers of copies already placed for the video. We will apply this general method to all multiple video placement policies in this section to obtain the optimal video placement subject to some storage constraint.

**Algorithm 3** General algorithm for multiple video placement

1: Perform single video analysis on each video to obtain the set of placement performance curves for all videos
2: **while** storage constraint not violated **do**
3: - find a video with the highest marginal performance gain given copies of videos previously stored
4: - store a copy of the video in the cache position obtained from the single video analysis
5: **end while**

## 6.2 Fixed Whole Storage Multiple Video Placement Policy

From the single video analysis for fixed whole storage placement policy, recall that $\alpha(m)$ is the probability video $m$ is present in a typical cache. In general, we can write $\alpha(m) \cdot |H| = k + \theta$, where $k$ is an integer and $0 \leq \theta < 1$, and let $X_m$ be the minimum variance integer valued random variable with mean $\alpha(m) \cdot |H|$. Specifically, $P\{X_m = k\} = 1 - \theta$ and $P\{X_m = k+1\} = \theta$. Given $X_m$, the video is assumed to be placed into a set of caches of cardinality $X_m$, with all $\binom{|H|}{X_m}$ possibilities having equal probability.

The procedure of selecting $\alpha(m)$ is nonadaptive in the sense that the particular set of caches in which the video is placed is independent of which caches the peers are connected to. If $\{\alpha(m) : m \in M\}$ is selected based on the popularities of videos, the procedure is adaptive in the sense that the total number of peers served by the whole system can be maximized. Recall the set of videos, $M$, follows a decreasing Zipf popularity distribution, $p(m)$ for each $m \in M$, and $p_{miss}(m)$ is the probability a peer is not connected to a cache storing video $m$, given by $p_{miss}(m) = \frac{\binom{|H|-C_m}{L}}{\binom{|H|}{L}}$.

Now consider multiple videos with Proposition 4.1.1; we have the following upper bound on $p_{miss}$, the average probability a peer is not connected to a cache storing the requested video:

$$p_{miss} \triangleq \sum_{m=1}^{|M|} p(m) \cdot p_{miss}(m) \leq \sum_{m=1}^{|M|} p(m) \cdot (1 - \alpha(m))^L \tag{6.1}$$

Using the upper bound on $p_{miss}$, Algorithm 3 can be carried out analytically. Suppose each cache can store $K$ copies of videos. To minimize the upper bound on $p_{miss}$, we select $(\alpha(m) : m \in M)$ to minimize $\sum_{m \in M} p(m)(1 - \alpha(m))^L$ subject to the constraint $\sum_{m=1}^{|M|} \alpha(m) = K$. This convex optimization problem can be solved using the Karush-Kuhn-Tucker conditions with a Lagrange multiplier for the sum constraint, yielding:

$$\alpha(m) \left(1 - \frac{c}{(p(m))^{\frac{1}{L-1}}}\right)_+ \qquad (6.2)$$

$$\text{where c is chosen so } \sum_{m=1}^{|M|} \alpha(m) = K.$$

A binary bisection algorithm can be used to quickly find $c$ numerically. As a result, given each cache can store $K$ copies of videos, Equation (6.2) gives the optimal probability any video is present in a typical cache. Any fixed whole storage placement for multiple videos with empirical probability of each video $m$ present in the caches equal to $\alpha(m)$ would serve the maximum expected number of peers for the fixed whole storage placement policy.

The same performance is obtained when the uniform per-cache storage constraint is replaced by the total storage constraint. This is because the cache-peer connections are random, so the performance does not depend on which cache a video is stored in.

## 6.3 Fixed Fractional Storage Multiple Video Placement Policy

From the single video analysis for fixed fractional storage placement policy, recall that each cache stores a fractional copy of uniform size, so the marginal performance gain is constant for the placement of any video and is proportional to the popularity distribution. Applying Algorithm 3, given each cache can store $K$ copies of videos and each peer is connected to $L$ distinct caches, the placement of a fraction $\frac{1}{L}$ of each of the $K \cdot L$ most popular videos in each cache would serve the maximum number of peers for the fixed fractional storage placement policy.

The uniform per-cache storage constraint is equivalent to the total cache storage constraint because of the uniform size of fractional copies stored in each cache.

## 6.4 Adaptive Whole Storage Multiple Video Placement Policy

From the single video analysis for the adaptive whole storage placement policy, we see that finding $H_m$ caches that are connected to the maximum number of peers requesting video $m$ is a set cover problem which may require exhaustive search to solve, and the placement problem would be more difficult for multiple videos. Therefore, it is preferable to extend the single video greedy peeling algorithm, Algorithm 1, to a multiple video greedy algorithm, which is in the exact form of Algorithm 3.

Approximately the same performance can be obtained when the total storage constraint is replaced by the uniform per-cache storage constraint. Because the cache-peer connections are random and the Zipf popularity distribution is a heavy-tailed probability distribution, there are many different videos with the same marginal performance gain and the peers which are requesting these videos are connected to many different caches. Therefore, there is ample freedom to balance the per-cache storage of videos.

## 6.5 Adaptive Fractional Storage Multiple Video Placement Policy

From the single video analysis for the adaptive fractional storage placement policy, the primal-dual algorithm, Algorithm 2, can be extended to multiple videos by combining primal and dual variables for all videos. For each video $m$, the summation of prices of download rates in primal 2 represents the total demand of users for video $m$ in cache $h$. This is similar to finding the marginal performance gain in Algorithm 1 and increases the storage of video $m$ proportional to the marginal performance gain. The result gives the optimal placement [11].

The uniform per-cache and the total cache storage constraints can both be satisfied directly through the primal-dual algorithm by changing the dual variable on cache storages. Approximately the same performance can be obtained for the total storage constraint and the uniform per-cache storage constraint, because of the same reason for adaptive whole storage placement policy.

## 6.6 Algorithm for Hybrid Multiple Video Placement Policy

Finally, we can construct an extension of Algorithm 3 for a hybrid storage placement policy for the whole system. For popular videos, the policy uses the fixed fractional storage placement policy because of the law of large numbers approximation. For less popular videos, the policy uses adaptive whole storage placement policy because of randomness. The main advantage of the hybrid storage placement policy is computational efficiency and immediate algorithm convergence. The performance, numbers of peers served by the whole system, of this policy is also strictly better than the fixed fractional storage and adaptive whole storage placement policy. The algorithm for the hybrid storage placement policy is as follows:

**Algorithm 4** Algorithm for hybrid multiple video placement policy

1: Separate video catalog into two categories, the popular videos and less popular videos
2: For popular videos, select fixed fractional storage placement policy
3: For less popular videos, select adaptive whole storage placement policy
4: Based on the choice of policy for each video, perform single video performance analysis for every video
5: Obtain a list of marginal performance gains of storing copies of any video constraint is satisfied
6: Store copies of videos with the highest performance gains in the list corresponding to the choice of their placement policies such that the total storage

## 6.7 Multiple Video Simulation Results

For a large system, we choose 40,000 peers, 50 caches, and 2000 videos following a 0.8 Zipf popularity distribution, where each peer is connected to 4 random caches selected uniformly, forming a random graph. The system's total storage constraint is 2.5 times the entire video catalog, which is 5000 copies. The placements of multiple videos according to adaptive fractional storage placement policy, adaptive whole storage placement policy, and fixed fractional storage placement policy are shown in Figure 6.1. The number of video copies stored in the system is plotted versus the videos listed in the order of decreasing popularity. The total numbers of peers served by the cache system are shown in Table 6.1 for each multiple video placement policy. The adaptive fractional storage placement policy yields the maximum number of peers that are served by the cache system without help from the server for multiple videos.

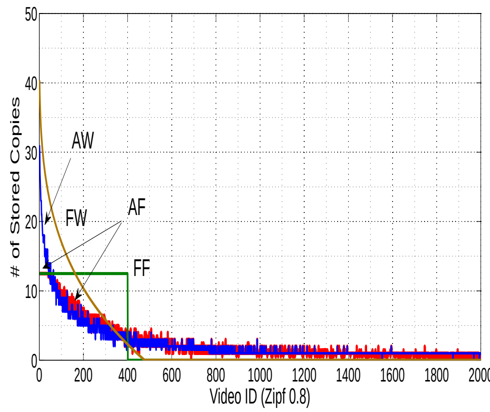


Figure 6.1: Video copies stored in a whole system of 50 caches, 40,000 peers and four random connections per peer

Table 6.1: Total number of peers served by the cache system

| Multiple video placement policy | Total # of peers served | Fraction of the optimal performance |
|---|---|---|
| Fixed whole storage | 21747 | 69.9% |
| Fixed fractional storage | 26348 | 84.6% |
| Adaptive whole storage | 29833 | 95.8% |
| Adaptive fractional storage | 31127 | 100% |

For the same large system and constraint, the hybrid storage placement policy stated in Algorithm 4 is shown in Figure 6.2, in which videos 1 to 100 belong to the popular videos category and video 101 to 2000 belong to the less popular category. The number of video copies stored in the system is plotted versus the videos in the order of decreasing popularity. The total number of peers served by the cache system is 30706, which is about 98.6% of the performance of the optimal policy - adaptive fractional storage placement policy.

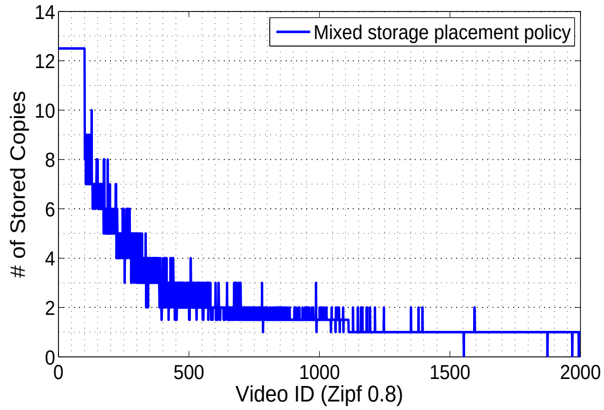


Figure 6.2: Video copies stored in a whole system of 50 caches, 40,000 peers and four random connections per peer

# CHAPTER 7

# CONCLUSION AND FUTURE WORK

In this thesis, we studied the content placement problem for cache delivery VoD systems. Instead of performing content placement analysis on the whole system with multiple videos, we approached the content placement problem by analyzing the decoupled systems with only a single video. With the insights gained from single video placement policy analysis, we returned to the original content placement problem by constructing a general method for the four placement policies for multiple videos, which places copies of videos so as to maximizes the total expected download rate for all users from the caches. We provided analytical and simulation results that answer the key question of how good is fractional storage or adaptive placement. Finally, based on these results, we proposed an extension of the general method for a hybrid storage placement policy for the whole system, which is a simple suboptimal alternative to the optimal content placement policy.

In the future, we will look for a rigorous method of optimally applying the hybrid multiple video placement policy, specifically, how to separate the video catalogs into two categories so the better content placement policy chosen from the fixed fractional and the adaptive whole storage placement policies is applied.

# REFERENCES


[1] V. K. Adhikari, S. Jain, and Z. L. Zhang, "Where do you "tube"? uncovering youtube server selection strategy," in *Proc. 20th International Conference on Computer Communications and Networks*, Maui, Hawaii, July 2011, pp. 1–6.

[2] "Akama," 2013. [Online]. Available: http://www.akamai.com/

[3] "Pplive," 2013. [Online]. Available: http://www.pptv.com/

[4] J. Almeida, D. Eager, M. Vernon, and S. Wright, "Minimizing delivery cost in scalable streaming content distribution systems," *IEEE Transactions on Multimedia*, vol. 6, pp. 356–365, Apr. 2004.

[5] Y. Boufkhad, F. Mathieu, F. de Montgolfier, D. Perino, and L. Viennot, "Achievable catalog size in peer-to-peer video-on-demand systems," in *Proc. 7th International Conference on Peer-to-Peer Systems*, Tempa, Florida, Feb. 2008, pp. 4–4.

[6] B. Tan and L. Massoulié, "Brief announcement: adaptive content placement for peer-to-peer video-on-demand systems," in *Proc. 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, Zurich, Switzerland, July 2010, pp. 293–294.

[7] X. Zhou and C. Xu, "Efficient algorithms of video replication and placement on a cluster of streaming servers," *Journal of Network and Computer Applications*, vol. 30, pp. 515–540, Apr. 2007.

[8] N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis, "On the optimization of storage capacity allocation for content distribution," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 47, pp. 409–428, Feb. 2005.

[9] J. Wu and B. Li, "Keep cache replacement simple in peer-assisted vod systems," in *Proc. 28th Conference on Computer Communications*, Rio de Janeiro, Brazil, Apr. 2009, pp. 2591–2595.

[10] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. Ramakarishnan, "Optimal content placement for a large-scale vod system," in *Proc. 6th ACM International Conference on emerging Networking EXperiments and Technologies*, Philadelphia, PA, Dec. 2010, pp. 4–4.

[11] H. Zhang, "An optimized video-on-demand system: Theory, design and implementation," University of California, Berkeley, CA, Tech. Rep. UCB/EECS-2012-229, Dec. 2012.